UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/807,499	03/24/2004	David John Butcher	550-540	4256

23117 7590 01/07/2009

NIXON & VANDERHYE, PC
901 NORTH GLEBE ROAD, 11TH FLOOR
ARLINGTON, VA 22203

EXAMINER	
LI, AIMEE J	

ART UNIT	PAPER NUMBER
2183	

MAIL DATE	DELIVERY MODE
01/07/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/807,499
Filing Date: March 24, 2004
Appellant(s): BUTCHER ET AL.

____
Stanley C. Spooner (Reg. No.27,393)
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 21 July 2008 appealing from the Office actions mailed 15 October 2007 and 28 March 2008.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6484314	Ishizaki et al.	11-2002
5727227	Schmidt et al.	3-1998

Hennessy, John L. and Patterson, David A. Computer Organization and Design the Hardware/Software Interface. Second Edition. San Francisco, CA: Morgan Kaufmann Publishers, Inc., ©1998. Pages 410-416.

Hyde, Randall. The Art of Assembly Language Programming. ©1996. Chapters: Table of Contents; Chapter 6 Part 2; and Chapter 6 Part 5.
http:llwww.arl.wustl.edul~lockwoodlclasslcs3061bookslartofasmltoc.html

Wikipedia, the free encyclopedia. ©2003. www.wikipedia.com search term: protected mode

Free On-Line Dictionary of Computing. FOLDOC. ©1995-1999. www.foldoc.org
search terms: central processing unit; arithmetic and logic unit; control unit; machine cycle; and operating system

Tanenbaum, Andrew S. Structured Computer Organization. Second Edition. Englewood Cliffs, NJ: Prentice-Hall, Inc., ©1984. Pages 10-12.

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claims 1-5, 7, 9-11, 13-17, 19, 21-23, 25-29, 31, 33-35, 37-41, 43, and 45-47 are rejected under 35 U.S.C. 103(a) as being unpatentable over Ishizaki et al., U.S. Patent Number 6,484,314 (herein referred to as Ishizaki) in view of Hennessy and Patterson's Computer Organization and Design: The Hardware/Software Interface ©1998 (herein referred to as Hennessy).

Referring to claims 1, 13, 25, and 37, taking claim 1 as exemplary, Ishizaki has taught apparatus for processing data comprising:

Processing logic operable to perform data processing operations (Ishizaki column 6, lines 11-29; Figure 4; and Figure 5). In regards to Ishizaki, the processing logic is inherent to

the CPU of Ishizaki. See FOLDOC "central processing unit" ©1998 and "arithmetic and logic unit" ©1995 for more information.

An instruction decoder for decoding program instructions to control said processing logic to perform data processing operations specified by said program instructions (Ishizaki column 6, lines 11-29; Figure 4; and Figure 5). In regards to Ishizaki, the decoder is inherent to the CPU of Ishizaki. See FOLDOC "central processing unit" ©1998, "control unit" ©1995, and "machine cycle" ©1995 for more information.

Wherein said instruction decoder, in response to a compare and branch instruction (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; and column 2, lines 4-11), comprises a decoder for:

Performing a comparison between a first value stored in a first register and a second value stored in a second register (Ishizaki column 4, line 48 to column 5, line 34 and column 5, line 51 to column 6, line 10);

For determining a target branch address from a pre-programmed stored value (Ishizaki column 4, line 48 to column 5, line 34 and column 5, line 51 to column 6, line 10); and

For branching to a sub-routine at said target branch address in dependence upon a result of said comparison (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; column 2, lines 4-11; column 4, line 48 to column 5, line 34 and column 5, line 51 to column 6, line 10).

Ishizaki has not explicitly taught for copying, in dependence upon a result of said comparison, a program counter value to a third register, and for determining a target branch address from said program counter value (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; column 2, lines 4-11; column 4, line 48 to column 5, line 34 and column 5, line 51 to column 6, line 10).

However, Ishizaki has taught branching on an exception and exception handling, but not the specifics on how exception handling affects the program counter. Hennessy has explicitly taught copying, in dependence upon a result of said comparison, a program counter value to a third register, and for determining a target branch address from said program counter value (Hennessy pages 411-413, section How Exceptions are Handled). In regards to Hennessy, the program counter of the instruction causing the exception is stored in the EPC and, as Hennessy states on page 412 "The operating system knows the reason for the exception by the address at which it is initiated." This means that the address of the instruction causing the exception is needed to determine why the exception was initiated so that the proper exception handler is accessed. A person of ordinary skill in the art at the time the invention was made, and as taught by Hennessy, would have recognized that copying the program counter value and determining a target branch address from the program counter value allows the operating system to take the appropriate action to report and correct the error then restart execution of the program (Hennessy page 411). Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate copying the program counter value and determining the branch target address from the program counter value to ensure the exception handler takes the appropriate action to report and correct the error and restart program execution when the exception is handled.

Claims 13, 25, and 37 contain similar limitations to claim 1 and are rejected for similar reasons. The claims differ from claim 1 in that claim 13 is a method and claims 25 and 37 are computer program products.

Regarding to claims 2, 14, 26, and 38, Ishizaki in view of Hennessy has taught, taking claim 2 as exemplary, apparatus as claimed in claim 1, wherein said instruction is an array bounds checking instruction and said sub-routine is an array bounds exception handling routine (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; column 2, lines 4-11; column 4, line 48 to column 5, line 34 and column 5, line 51 to column 6, line 10). Claims 14, 26, and 38 contain similar limitations to claim 2 and are rejected for similar reasons.

Regarding claims 3, 15, 27, and 38, Ishizaki in view of Hennessy has taught, taking claim 3 as exemplary, apparatus as claimed in claim 1, wherein at least one of said first register and said second register are specified within said compare and branch instruction (Ishizaki column 4, line 48 to column 5, line 34 and column 5, line 51 to column 6, line 10). Claims 15, 27, and 38 contain similar limitations to claim 3 and are rejected for similar reasons.

Regarding claims 4, 16, 28, and 40, Ishizaki has taught, taking claim 4 as exemplary, apparatus as claimed in claim 2, wherein said first value is a reference value specifying an array size and said second value is a test value determined from a decoded program instruction (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; column 2, lines 4-11; column 4, line 48 to column 5, line 34 and column 5, line 51 to column 6, line 10). Claims 16, 28, and 40 contain similar limitations to claim 4 and are rejected for similar reasons.

Regarding claims 5, 17, 29, and 41, Ishizaki in view of Hennessy has taught, taking claim 5 as exemplary, apparatus as claimed in claim 4, wherein said comparison determines whether said reference value is greater than or equal to said test value (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; column 2, lines 4-11; column 4, line 48 to column 5, line 34 and column 5, line 51

to column 6, line 10). Claims 17, 29, and 41 contain similar limitations to claim 5 and are rejected for similar reasons.

Regarding claims 7, 19, 31, and 43, Ishizaki has taught, taking claim 7 as exemplary, apparatus as claimed in claim 2, wherein said branching operation comprises copying a pointer to said array bounds exception handling routine into a register specifying a next program instruction (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; and column 2, lines 4-11). Claims 19, 31, and 43 contain similar limitations to claim 7 and are rejected for similar reasons.

Regarding claims 9, 21, 33, and 45, Ishizaki in view of Hennessy has taught, taking claim 9 as exemplary, apparatus as claimed in claim 1, wherein said compare and branch instruction is executed within a single processing cycle of said data processing apparatus when the branch is not taken (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; column 2, lines 4-11; column 4, line 48 to column 5, line 34 and column 5, line 51 to column 6, line 10). Claims 21, 33, and 48 contain similar limitations to claim 9 and are rejected for similar reasons.

Regarding claims 10, 22, 34, and 46, Ishizaki in view of Hennessy has taught, taking claim 10 as exemplary, apparatus as claimed in claim 1, wherein said instruction decoder is operable to decode translated platform-independent program instructions (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; and column 2, lines 4-11). Claims 22, 34, and 46 contain similar limitations to claim 10 and are rejected for similar reasons.

Regarding claims 11, 23, 35, and 47, Ishizaki in view of Hennessy has taught, taking claim 11 as exemplary, apparatus as claimed in claim 10, wherein said platform independent program instructions are one of:

Java bytecodes (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; and column 2, lines 4-11);

.net bytecodes;

MSIL bytecodes; and

CIL bytecodes.

Claims 22, 34, and 46 contain similar limitations to claim 10 and are rejected for similar reasons.

Claims 6, 18, 30, and 42 are rejected under 35 U.S.C. 103(a) as being unpatentable over Ishizaki in view of Hennessy, as applied to claims 4, 16, 28, and 40 above, in view of "The Art of Assembly Programming" ©30 September 1996 (herein referred to as Assembly Programming).

Taking claim 6 as exemplary, Ishizaki has not taught apparatus as claimed in claim 4, wherein said result of said comparison is determined from a carry flag value and zero flag value. Assembly Programming has taught wherein said result of said comparison is determined from a carry flag value and zero flag value (Assembly Programming Sections 6.5.3 and 6.9.4). Ishizaki has taught in column 5, line 17 that a greater than or equal to compare and branching instruction is performed. However, Ishizaki has not taught explicitly how the compare functions, e.g. how the results are determined, and how the compare instruction directly affects the jump function. Assembly Programming has explicitly taught a method for the compare instruction, which only sets the flags register (Assembly Programming Section 6.5.3), and that the compare instruction flag results directly influence the conditional jumps (Assembly Programming Section 6.9.4). A person of ordinary skill in the art at the time the invention was made would have recognized that the compare and jumps of Assembly Programming implements the compare and jumps without using too much memory, since it does not need to store the subtraction results. Therefore, it

would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the compare and jumps of Assembly Programming in the device of Ishizaki. Claims 18, 30, and 42 contain similar limitations to claim 6 and are rejected for similar reasons.

Claims 8, 20, 32, and 44 are rejected under 35 U.S.C. 103(a) as being unpatentable over Ishizaki in view of Hennessy, as applied to claims 1, 13, 25, and 37 above, in view of Schmidt et al., U.S. Patent Number 5,727,227 (herein referred to as Schmidt). Taking claim 8 as exemplary, Ishizaki has not explicitly taught apparatus as claimed in claim 1, wherein said data processing apparatus comprises a co-processor and said pre-programmed stored value is read from a register of said co-processor. Schmidt has taught wherein said data processing apparatus comprises a co-processor and said pre-programmed stored value is read from a register of said co-processor (Schmidt column 3, line 45 to column 4, line 49; Figure 1; and Figure 3). Ishizaki has taught in column 6, lines 28-29 that a multi-CPU configuration may be used, but has not taught the specific functions of each CPU or their purpose in the system. Schmidt has explicitly taught the functions of each CPU in a multi-CPU system and their purpose in the overall system. A person of ordinary skill in the art at the time the invention was made, and as taught by Schmidt, would have recognized that the co-processor system of Schmidt reduces the amount of time needed to process interrupt/exception routines, thereby improving the speed of the system (Schmidt column 3, lines 33-43). Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the co-processor of Schmidt in the device of Ishizaki to improve processor speed. Claims 20, 32, and 44 contain similar limitations to claim 8 and are rejected for similar reasons.

Claims 12, 24, 36, and 48 are rejected under 35 U.S.C. 103(a) as being unpatentable over Ishizaki in view of Hennessy, as applied to claims 1, 13, 25, and 37 above, in view of Wikipedia term "Protected Mode" ©October 2003 (herein referred to as Wikipedia). Taking claim 9 as exemplary, Ishizaki has taught an apparatus as claimed in claim 1, said data processing apparatus remains in a user mode during execution of said compare and branch instruction (Ishizaki Abstract, lines 16-20; column 1, lines 13-42; column 2, lines 4-11; column 4, line 48 to column 5, line 34 and column 5, line 51 to column 6, line 10). Ishizaki has not taught wherein said data processing apparatus is operable in a user mode and a privileged mode. Wikipedia has taught wherein said data processing apparatus is operable in a user mode and a privileged mode (Wikipedia term "Protected mode"). A person of ordinary skill in the art at the time the invention was made would have recognized that protected mode does not allow other tasks to see the current tasks memory, thereby making multi-tasking more stable (Wikipedia term "Protected mode"). Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the modes of Wikipedia in the device of Ishizaki to make the system more stable for multi-tasking. Claims 24, 36, and 48 contain similar limitations to claim 12 and are rejected for similar reasons.

Below is a table showing a more detailed mapping and explanation between the exemplary independent claim 1 and Ishizaki in view of Hennessy.

Instant Application	Prior Art
Claim 1	Ishizaki et al., U.S. Patent Number 6,484,314 (herein referred to as Ishizaki) in view of Hennessy and Patterson's <u>Computer Organization and Design: The Hardware/Software Interface</u> ©1998 (herein referred to as Hennessy)
Apparatus for processing data comprising:	

processing logic operable to perform data processing operations; and	Ishizaki column 6, lines 11-29 "…A system **100** comprises a central processing apparatus (CPU) **1**…" Figure 4 Figure 5 In regards to Ishizaki, the processing logic is inherent to the CPU of Ishizaki. See FOLDOC "central processing unit" ©1998 and "arithmetic and logic unit" ©1995 for more information.
an instruction decoder for decoding program instructions to control said processing logic to perform data processing operations specified by said program instructions,	Ishizaki column 6, lines 11-29 "…A system **100** comprises a central processing apparatus (CPU) **1**…" Figure 4 Figure 5 In regards to Ishizaki, the decoder is inherent to the CPU of Ishizaki. See FOLDOC "central processing unit" ©1998, "control unit" ©1995, and "machine cycle" ©1995 for more information.
wherein said instruction decoder, in response to a compare and branch instruction, comprises a decoder for:	Ishizaki Abstract, lines 16-20 "An exception checking instruction is an instruction for determining whether an exception condition is established, and for branching to an exception handler when the exception condition is established." column 1, lines 13-42 "…the detection of exceptions…" column 2, lines 4-11 "…a comparison instruction or a branching instruction can be used to indicate the type of exception…"
performing a comparison between a first value stored in a first register and a second value stored in a second register;	Ishizaki column 4, line 48 to column 5, line 34 "…The tw/twi instructions are instructions that perform comparison and branching at the same time, so that when a condition described in the operand is established, program control branches to a handler designated by the OS…" column 5, line 51 to column 6, line 10 "…The implementation of an instruction prepared to provide high speed comparison and branching…"
copying, in dependence upon a result of said comparison, a program counter value to a third	Ishizaki Abstract, lines 16-20 "An exception checking instruction is an instruction for determining whether

register; determining a target branch address from said program counter value	an exception condition is established, and for branching to an exception handler when the exception condition is established." column 1, lines 13-42 "…the detection of exceptions…" column 2, lines 4-11 "…a comparison instruction or a branching instruction can be used to indicate the type of exception…" column 4, line 48 to column 5, line 34 "…The tw/twi instructions are instructions that perform comparison and branching at the same time, so that when a condition described in the operand is established, program control branches to a handler designated by the OS…" column 5, line 51 to column 6, line 10 "…The implementation of an instruction prepared to provide high speed comparison and branching…" Figure 1 Figure 2 Ishizaki has not explicitly taught these limitations, however, in the cited sections above Ishizaki has taught branching on an exception and exception handling, but not the specifics on how exception handling affects the program counter value, i.e. the address of the current instruction to be executed. Hennessy pages 411-413, section **How Exceptions Are Handled** "…The basic action that the machine must perform when an exception occurs is to save the address of the offending instruction in the exception program counter (EPC) and then transfer control to the operating system at some specified address…" Hennessy has explicitly taught these limitations in the cited sections above. Hennessy teaches that the program counter of the instruction causing the exception is stored in the EPC and, as Hennessy states on page 412 "The operating system knows the reason for the exception by the address at which it is initiated." This means that the address of the instruction causing the exception is needed to determine why the exception was initiated so that the

	proper exception handler is accessed. A person of ordinary skill in the art at the time the invention was made, and as taught by Hennessy, would have recognized that copying the program counter value and determining a target branch address from the program counter value allows the operating system to take the appropriate action to report and correct the error then restart execution of the program (Hennessy page 411). Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate copying the program counter value and determining the branch target address from the program counter value to ensure the exception handler takes the appropriate action to report and correct the error and restart program execution when the exception is handled.
determining a target branch address from a pre-programmed stored value; and	Ishizaki column 4, line 48 to column 5, line 34 "…The tw/twi instructions are instructions that perform comparison and branching at the same time, so that when a condition described in the operand is established, program control branches to a handler designated by the OS…" column 5, line 51 to column 6, line 10 "…The implementation of an instruction prepared to provide high speed comparison and branching…"
branching to a sub-routine at said target branch address in dependence upon a result of said comparison.	Ishizaki Abstract, lines 16-20 "An exception checking instruction is an instruction for determining whether an exception condition is established, and for branching to an exception handler when the exception condition is established." column 1, lines 13-42 "…the detection of exceptions…" column 2, lines 4-11 "…a comparison instruction or a branching instruction can be used to indicate the type of exception…" column 4, line 48 to column 5, line 34 "…The tw/twi instructions are instructions that perform comparison and branching at the same time, so that when a condition described in the operand is established, program control branches to a handler designated by the OS…"

	column 5, line 51 to column 6, line 10 "…The implementation of an instruction prepared to provide high speed comparison and branching…"

The table below shows a more detailed mapping between the independent claims.

Claim 1	Claim 13	Claim 25	Claim 37
Apparatus for processing data comprising:	A method of processing data with an apparatus for processing data having	A computer program product comprising a computer-readable storage medium including a computer program operable to control an apparatus for processing data having	A computer program product comprising a computer-readable storage medium including a computer program operable to translate non-native program instructions to form native program instructions directly decodable by an apparatus for processing data having
processing logic operable to perform data processing operations; and	processing logic operable to perform data processing operations and	processing logic operable to perform data processing operations and	processing logic operable to perform data processing operations and
an instruction decoder for decoding program instructions to control said processing logic to perform data processing operations specified by said program instructions, wherein said instruction decoder, in response to a compare and branch instruction,	an instruction decoder operable to decode a compare and branch instruction to control said processing logic to perform data processing operations specified by said program instructions, said method comprising the steps of	an instruction decoder operable to decode a compare and branch instruction to control said processing logic to perform data processing operations specified by said program instructions, said computer program comprising the steps of:	an instruction decoder operable to decode a compare and branch instruction to control said processing logic to perform data processing operations specified by said program instructions, said native program instructions comprising:

comprises a decoder for:			
(i) performing a comparison between a first value stored in a first register and a second value stored in a second register;	(i) performing a comparison between a first value stored in a first register and a second value stored in a second register;	(i) performing a comparison between a first value stored in a first register and a second value stored in a second register;	(i) performing a comparison between a first value stored in a first register and a second value stored in a second register;
(ii) copying, in dependence upon a result of said comparison, a program counter value to a third register;	(ii) copying a program counter value, in dependence upon said comparison, to a third register;	(ii) copying a program counter value, in dependence upon said comparison, to a third register;	(ii) copying a program counter value, in dependence upon said comparison, to a third register;
(iii) determining a target branch address from a pre-programmed stored value and said program counter value; and	(iii) determining a target branch address from a pre-programmed stored value and said program counter value; and	(iii) determining a target branch address from a pre-programmed stored value; and	(iii) determining a target branch address from a pre-programmed stored value; and
(iv) branching to a sub-routine at said target branch address in dependence upon a result of said comparison.	(iv) branching to a sub-routine at said target branch address in dependence upon a result of said comparison.	(iv) branching to a sub-routine at said target branch address in dependence upon a result of said comparison.	(iv) branching to a sub-routine at said target branch address in dependence upon a result of said comparison.

(10) Response to Argument

In general Applicants' arguments attempt to show the invalidity of Ishizaki in view of Hennessy.

Specifically, on pages 10-11and 15-16 Applicants' first argue in essence

> "…Hennessy cannot teach the claimed 'copying, in dependence upon a result of said comparison,…' or the 'branching…in dependence upon a result of said comparison.'"

Applicants' arguments further attempt to support this conclusion by stating "In Hennessy, the program counter value (address of the offending instruction) is copied to an exception program

counter not in response to a comparison result…, but rather, in response to an exception." However, Applicants' arguments have failed to take into consideration the teachings of Ishizaki and its application to the other limitations in the claims, specifically the limitations referring to the comparison itself. Specifically, as cited in rejection above, Ishizaki has taught that an exception is a conditional branch. In order to determine if an exception condition occurs, a comparison and branch are made using the tw/twi instructions. The tw/twi instructions compare to see if a specific condition exists and branches accordingly. As explained further in the rejection above, Ishizaki has taught comparing and branching to a specific handler, but not the specific details of how the branching to the specific handler is performed, i.e. how branching to an exception handler affects the program counter value and other elements within the processor. Hennessy was relied upon to teach these specifics. Consequently, in the combination of Ishizaki in view of Hennessy, when Hennessy teaches performing the copying and determining steps in response to an exception, it is in response to a compare condition and branch exception, such as that taught in Ishizaki. Also, Examiner notes that secondary references, such as Assembly Programming, Schmidt or Wikipedia were provided to show that the steps of copying and determining are readily known in the art and common practice.

Applicants' second argument on pages 12 and 16-17 argues in essence

> "The Examiner fails to provide any reason or motivation for combining references…the sort of 'conclusory' statement that the Supreme court has held is insufficient to establish a case of obviousness."

As quoted from the rejection (copied above), the Examiner states "it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate copying the

program counter value and determining the branch target address from the program counter value **to ensure the exception handler takes the appropriate action to report and correct the error and restart program execution when the exception is handled** (emphasis added)." Also, the Examiner stated in the previous sentence "A person of ordinary skill in the art at the time the invention was made, and as taught by Hennessy, would have recognized that copying the program counter value and determining a target branch address from the program counter value **allows the operating system to take the appropriate action to report and correct the error then restart execution of the program (Hennessy page 411)** (emphasis added)." The highlighted language is the motivation to combine. These statements are the reason a person of ordinary skill in the art would have combined Hennessy into Ishizaki. Simply put, a person of ordinary skill in the art at the time the invention was made, and as taught by Hennessy, would have combined Hennessy into Ishizaki to make sure that appropriate action is taken to correct the error and that program execution will restart from the correct address after the exception is handled. As is known in the art, and Ishizaki and Hennessy have both taught, exceptions are errors in the program that must be handled before correct program execution can continue. By performing the copying and determining steps of Hennessy, a person of ordinary skill in the art ensures that the error is corrected and the program restarted when the error has been corrected.

Applicants' third argument on pages 12-13 and 17 argues in essence

"The Examiner fails to recognize that she has misunderstood the Hennessy reference." It is unclear to the Examiner how the motivation provided (see response to the second arguments above) illustrates a misunderstanding of Hennessy. Ishizaki has taught a compare and branch

instruction for determining types of exceptions and branching to the appropriate exception handler in order to correct the exception, but not the specific details of branching to the appropriate exception handler. Hennessy teaches that, in response to an exception, copying the program counter value and determining the instruction address of the appropriate exception handler ensures that the appropriate exception handler is executed and the program can restart from the appropriate instruction address.

Applicants' fourth argument on pages 13-14 and 17-18 argues in essence

> "…Ishizaki would clearly lead one of ordinary skill in the art away from the claims 'copying' and 'branching' steps…".

Ishizaki does not clearly teach away from the "copying" and "branching" steps. As explained in the rejection above, Ishizaki teaches a single instruction for comparing to determine the type of exception and branching to the appropriate exception handler in general and does not go into the details of how the comparing and branching affects specific values and elements within the CPU, such as the program counter and the program counter value. Hennessy has taught these details. There is nothing within Ishizaki that even suggests that Ishizaki would not perform the "copying" and "branching" steps.

Applicants' fifth argument on pages 14-15 and 18 argues in essence

> "The Examiner fails to recognize that Ishizaki and Hennessy are mutually incompatible…Ishizaki, like the present invention, relates to the handling of '**<u>software exceptions</u>**,'…Hennessy relates to the handling of **<u>CPU exceptions</u>** whereby an unexpected event within the processor is analyzed and dealt with by the operating system…"

First, the operating system (OS) is software (please see the accompanying definition of "operating system" from FOLDOC ©1999), so, by Applicants' own arguments, Hennessy relates to software exceptions, since the operating system analyzes and deals with its exceptions. Also, Ishizaki has taught that "…when a condition described in the operand is established, program control branches to a handler designated by the OS (Ishizaki column 5, lines 32-34)." This means that the OS is involved with the handling of software exceptions. Second, Applicants' argument attempts to support its position by stating "the Examiner does admit that Hennessy' exception is 'an unexpected event **from within the processor**' (the CPU) and this is completely compatible with Appellant's position that Hennessy relates to **CPU exceptions** and not Ishizaki's **software exceptions**." The program causing a software exception operates within the processor, so it produces "an unexpected event from within the processor". Third, Applicants' argument attempts to further support its position by arguing "…she mistakenly asserts that MIPS is a software language (when in fact it is a RISC architecture…) and doesn't suggest a software language…" Applicants' are correct that MIPS is a specific RISC architecture, but every specific architecture has its own instruction set, i.e. software language. This instruction set, while having similar operations to other languages, such as add, subtract, compare, etc., has its own specific representation within the MIPS architecture. Hence, MIPS has its own software language, since it has its own individual instruction set not recognized by RISC architectures outside the MIPS family. Fourth, Applicants' arguments seem to be suggesting that Ishizaki and Hennessy are incompatible because Ishizaki is for software while Hennessy is for hardware, i.e. CPU. Whether the exception handling is in software or hardware, it does not matter, since they are functionally equivalent. The choice between software and hardware is a design choice

(please see the accompanying section from Tanenbaum's Structured Computer Organization ©1984 for more information).

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/Aimee J Li/

Primary Examiner, Art Unit 2183

Conferees:

/Kevin L Ellis/
Acting SPE of Art Unit 2187

/Eddie P Chan/
Supervisory Patent Examiner, Art Unit 2183